System requirement specification (SRS)

Name: Md. Ar-Raji Billah **Email: asirbillah@gmail.com** **Phone: +8801905651233**

1. Introduction

Weight indicates a lot about a person i.e. lifestyle, diet, health conditions, effect/ side effect of medicines, exercise habits etc. A US company decides to estimate the weight of human subjects with the help of efficient computer vision based algorithms which would work on cheap smartphones. They want to develop such a solution for their US clients to help medical professionals recommend proper diet and medicines during this obesity epidemic. This SRS describes the exploration of solution space, requirements, methodology etc. for building an effective human weight measurement system.

2. Ideation of Solution Space

The weight measurement of human subjects from images or videos is a trending research topic. But still there are only a few related works which we can consider relevant to discuss here.

2.1 Possible approaches

For all possible ways to solve this problem, the primary task is identical, which is, face-localization from video. As weight is a continuous value, the machine learning based prediction model should be a regression model instead of a classification model. According to the relevant papers, the possible approaches to develop the weight predictor are:

1. Calculate the BMI: Calculating BMI (Body Mass Index) is an ideal way to measure a person's weight related to his/ her height. [1]
2. Estimate the weight from full body contour: By proper image thresholding and segmentation methods, the full body contour can be extracted from which weight can be measured. [2]
3. Predict the weight from face or body landmarks: A system can be developed by generating face and body landmarks. [3]
4. Estimate the weight from only face image: This seems difficult but actually it is simple and outperforms many existing systems as there are necessary datasets and pretrained facial feature extraction systems available. [1]

2.2 Requirements

The user should be prompted to open the phone camera. Upon opening the camera, if the user starts capturing a video, the system will generate a bounding box around all human bodies in the video. Atleast, it will generate a bounding box around the faces of the front-facing humans. The system will predict the weight and display it to the user when the user touches the “predict” button. These are the functional requirements. The non-functional requirements are security, performance, capacity, reliability etc.

3. Methodology

3.1 Overview

The flow diagram of the methodology is shown below:

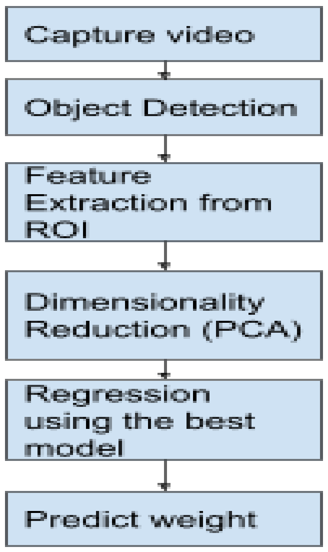
Capture video
Object Detection
Feature Extraction from ROI
Dimensionality Reduction (PCA)
Regression using the best model
Predict weight

Fig: Methodology Diagram

3.2 Dataset

For the experiments, we have chosen the ‘VIP-attribute dataset’ introduced by Dantcheva et al. which consists of 1026 samples (face pictures of 513 male and 513 female celebrities). These samples are front-facing face images and they have various image resolution and quality, expression and illumination. [1]

3.3 Object Detection

The first task is to extract face objects from videos. Sliding windows, Haar-cascade etc. should be chosen only when there are tight hardware constraints. In terms of performance, they are no match for comparatively new object detection algorithms. State of the art algorithms for object detection are:

1. MTCNN
2. YoloV3
3. SSD etc.

MTCNN is a popular face detection framework with better accuracy than the haar-cascade. The YoloV3 is very fast and efficient but it can be computationally expensive.

3.4 Feature Extraction

FaceNet algorithm is used to learn facial feature representations or embedding from pixels. It was trained using the ResNet pretrained CNN model. For the simplification of the regression or classification task, the embedding vector is fed to a machine learning model. The feature vector dimension is 128. For computational feasibility, the vector dimension is reduced using PCA (Principal Component Analysis) in this experiment.

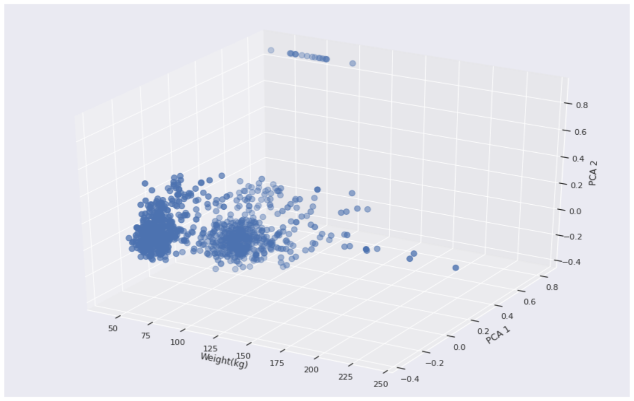

Fig: Dimensionality Reduction using PCA

3.5 Regression models

The regression models I have chosen for this experiments are:

Simple Linear Regression: Mean Square Error (MSE) for the simple linear eq. y=wx+b can be written as,

$MSE = (1/N).\sum(y_i - (w.x_i + b))^2$

Where, N is the number of examples, y_i is the class label and $(w.x_i + b)$ is the prediction.

Ridge Regression: It penalizes for large weight/ feature coefficient values by adding the $\Lambda.\sum w_j^2$ term with the linear regression MSE eq. Thus it controls overfitting.

Lasso Regression: It is the same as Ridge Regression except it uses the magnitude of w_j instead of square.

Random Forest Regression: It is a bagging ensemble technique that utilizes multiple decision trees. Both row sampling and feature sampling is done to the whole dataset and then passed to each decision tree with or without replacement. The gain values are calculated and the mean of outputs from all the decision trees are taken for regression.

Support Vector Regression (SVR): Support Vector Machine (SVM) not only tries to fit the data but also tries to maximize the margin between classes using dual formulation. Support Vector Regression tries to do regression in a non-linear space by using the kernel trick.

3.6 Implementation

Here the link for the code can be found: https://colab.research.google.com/drive/1kssvpz46lGveLxkRDaJYLUQo9bgJhqdI?usp=sharing

I have carried out the experiments using Google Colab GPU (for this task GPU is not mandatory). The following libraries were utilized:

1. Dlib
2. Face-Recognition
3. Numpy
4. Pandas
5. Sklearn
6. Matplotlib
7. Seaborn

3.7 Deployment

I have saved the best model using the sklearn joblib library which is a necessary step to integrate the model to android app. Instructions: https://www.youtube.com/watch?v=fNbxSXi0OkA&list=LL&index=3

4. Result Analysis

Performance of the weight prediction models is visualized in the following table:

Model	**MSE**	**Variance**	**Avg Error**	**Accuracy**
Linear Regression	0.03	0.60	0.1232	97.15%
Ridge Regression	0.03	0.61	0.1146	97.35%
Random Forest	0.03	0.59	0.1138	97.38%
SVR	0.03	0.63	0.1060	97.57%
Lasso Regression	0.07	0.00	0.2102	95.10%

According to average error and accuracy, support vector regression is the best performing model. Apart from weight, the models were also trained to predict height and BMI (Body Mass Index) which can be calculated using this formula,

BMI = weight [kg] / height^2 [m^2]

```
Height: 1.85m
Weight: 89.49kg
Bmi: 24.51kg/m^2
```

Fig: Sample prediction

5. Conclusion

In this work, various aspects related to a human weight prediction system from single shot face images and steps to build the system have been discussed. Though in the implementation I have not done face detection and deployment to an android system, the feature extraction and training of the ML regression models provided me with important insights related to this task. I couldn't implement object detection due to scarcity of dataset. But the 'VIP-attribute dataset' showed a decent performance for all the regression models. SVR gained the lowest average error (0.1060) and the highest accuracy (97.57%). In the future, I may build a weight detection model based on body contour extraction or landmark generating procedure.

6. References

I. **Dantcheva, A., Bremond, F., & Bilinski, P. (2018, August). Show me your face and I will tell you your height, weight and body mass index. In *2018 24th International Conference on Pattern Recognition (ICPR)* (pp. 3555-3560). IEEE.**

II. **Jiang, M., & Guo, G. (2019). Body weight analysis from human body images. *IEEE Transactions on Information Forensics and Security*, *14*(10), 2676-2688.**

III. **Rohan Soneja, Prashanth S and R Aarthi, "Body Weight Estimation using 2D Body Image" International Journal of Advanced Computer Science and Applications(IJACSA), 12(4), 2021. http://dx.doi.org/10.14569/IJACSA.2021.0120440**